Stateless TCP/IP Protocol

Background of the Invention

The present invention relates generally to communication protocols and more particularly to a novel stateless TCP/IP protocol.

Communication of data over the Internet between a pair of devices in communication with each is partially accomplished through the use of several layers of protocols or protocol stack. Generally, the stack includes the various application, protocol and communication layers that generate, encapsulate and transmit the data. The encapsulation generally places various headers on the data generated by an application program at the sending device to form a succession of packets for transmission into the Internet. The protocol stack at the receiving device is used, in reverse order, to strip such headers upon receipt of the packets so that the data can be reassembled and acted upon by another application program at the receiving device.

Generally, the top layer of the protocol stack is the application layer, followed by, in turn, the transport layer and network layer. Following the network layer is the data link layer and finally the access method layer which places the packet "onto the wire."

In the application layer, a program executing in the sending device initiates communications intended for the receiving device using an application protocol, for example, the file transfer protocol or the simple mail transfer protocol. The application layer is responsible for obtaining the IP address of the receiving device, wherein the IP address includes the network address and the host station address. The

application program at the sending device identifies the program it wishes to communicate with at the receiving device by its socket. The socket is a combination of the IP address of the receiving device and the port assigned to the program. A port is a logical number assigned to every program. A corresponding socket also exists at the sending device. The connection between the two devices is defined by the sockets.

The transport layer, for example using the Transmission Control Protocol (TCP), establishes a connection between the sending device and the receiving device before any data can be transmitted. Once established, a maximum packet size (generally limited by the allowable packet size of the data link layer) is negotiated between the two devices. TCP then attaches a header onto each packet, wherein the header contains, *inter alia*, the source port assigned to the program running at the sending device and the destination port assigned to the program running at the receiving device, as well as a sequence number of the packet. The sequence number is parsed by the stack at the receiving device to generate an acknowledgment packet sent back to the sending device. TCP then hands over the packet to the IP layer along with the IP address of the destination node.

The IP layer accepts the packets from TCP and prepares the packets for the data link protocol layer below by turning the IP addresses into physical station addresses, known to those skilled in the art as MAC addresses, and fragmenting the packets, if necessary, into a required frame size. Generally, the IP protocol broadcasts the IP address onto the network and the machine with that IP address responds with its MAC address. IP outputs packets called "datagrams," and each datagram is prefixed with an IP header that contains the source IP address of the sending device and the destination IP address of the receiving device. The IP protocol is used to route the packets from network to network.

The IP datagrams are then handed over to the data link layer. The data link layer is responsible for reliable node to node transmission within a subnetwork.

As can generally be seen from the above example, a message developed by a program running at the first device goes down the protocol stack, over the wire, and then back up the stack on the receiving device. The counterpart protocols in the stack at the time receiving device unpackage the frames, datagrams and packets, and then deliver the transmitted data to the program running at the receiving device for processing. Many references exist describing in greater detail the use of sockets and stacks. However, of particular relevance herein, are the TCP/IP protocol layers.

The TCP/IP connection as described above relies upon the socket and stack being placed in memory for each of the sending and receiving devices. Accordingly, the TCP/IP connection relies upon a state at each of the sending and receiving devices. The states at each device govern the activity the receptive device is performing. Furthermore, these activities require significant use of the respective device's resources to process the packets as they move up and down the stacks. For example, for each packet to be sent the TCP and IP header information needs to be computed and determined, and for each packet received the TCP and IP header information needs to be parsed.

In order for a socket connection to be established, each of the devices must synchronize to each other's initial sequence number, the sequence number being another part of the TCP header. The synchronization is performed by an exchange of connection establishing segments, or packets, carrying a control bit "SYN" in the TCP header along with the initial sequence number. In any such packet wherein the control bit is set, the packet is commonly referred to as a SYN.

The synchronization requires each device to send its own initial sequence number and to receive a confirmation of it in an acknowledgment from the other device. The acknowledgment is also sent in a packet wherein another control bit, ACK, is set, and this packet is thus commonly referred to as an ACK. The confirming acknowledgment includes an acknowledgment number, also part of the TCP header, which is derived from the sequence number it is acknowledging. The SYN and ACK control bits may both be set in a single TCP header so that the resultant packet, called a SYN/ACK, performs both functions.

An example of such a synchronization exchange may be given as follows:

A--->B	SYN (SEQ=X, CTRL=SYN)
B--->A	SYN/ACK (SEQ=Y, ACK=X+1, CTRL=SYN,ACK)
A--->B	ACK (SEQ=X+1, ACK=Y+1, CTRL=ACK)

Initiating the connection, the first device, A, sends the SYN with a randomly generated initial sequence number of the first device, as indicated by SEQ=X, and the synchronization control bit set, as indicated by CTRL+SYN. Upon receiving the SYN, the second device, B, generates the SYN/ACK with a randomly generated initial sequence number of the second device, as indicated by SEQ+Y, an acknowledgment number acknowledging the first device's sequence number, as indicated by ACK=X+1, and both the synchronization and acknowledgment control bits set, as indicated by CTRL=SYN,ACK. Completing the connection, the first device then sends the ACK with an incremented sequence number, as indicated by SEQ=X+1, an acknowledgment number acknowledging the second device's sequence number, as indicated by ACK=Y+1, and the acknowledgment control bit set, as indicated by CTRL=ACK.

In the above example, the first device, A, and the second device, B, each went through a change of states. The first device, A, had an initial state, CLOSED, and the second device, B, had an initial state, LISTEN. The CLOSED state indicates the

the device is idle, doing nothing, whereas the LISTEN state indicates the device is waiting to receive SYN's. Upon the SYN being sent and received, the first device, A, then changed to a state, SYN-SENT, and the second device, B, changed to a state, SYN-RECEIVED, the description of each of these states being apparent from its acronym. When the SYN/ACK is sent and received, the first device, A, changes state to ESTABLISHED, indicating a connection can be established, and the second device, B, maintains the SYN-RECEIVED state. Finally, when the ACK is sent and received, the first device, A, is still in the ESTABLISHED state, and the second device, B, also goes to the ESTABLISHED state, completing the socket connection.

An exchange of data may now occur between the first device and the second device, using data packets which may also acknowledge receipt of the other device's data packet. The IP header of each packet also contains a length number, which indicates the total length of the packet including both the TCP/IP headers and the data or message. From the length number, the receiving device can determine the total number of units of data received, wherein such units may, for example, be bytes or octets. Each unit received corresponds to an increment of one sequence number. The received length number in conjunction with the received sequence number is then used to determine the ACK number to be sent, which indicates the next sequence number the receiving device expects to receive from the sending device.

When either device has no more data to send, it sets another control bit, FIN, in the TCP header, to generate a FIN packet, which also must be acknowledge as above. During this exchange, each device is also move through various other states, as is known. When each device has sent a FIN packet that becomes acknowledged, the connection is closed. The state of the two devices then revert to the initial closed and listen states described above.

As is readily apparent from the brief description of the TCP/IP protocol above, numerous computations need to be performed consuming significant system resources in the memory dependent, state oriented connection. Furthermore, the receiving device in the connection described above may typically be a server which needs to accept connections from many sending devices, which may typically be clients. Two important parameters for servers are their ability to accept new connections, typically measured by the rate at which the server may establish new connections, and the maximum number of connections that the server may maintain.

Accordingly, because of the resource intensive nature of TCP/IP connections, TCP/IP is disadvantageously limited within a testing environment to obtain empirical data relating to these parameters. For example, the processing time required at a server may not allow the maximum rate of establishing new connections to be obtained. Therefore, a need exists for a stateless TCP/IP protocol that is useful in the testing environment.

Summary of the Invention

An object of the present invention is to overcome one or more disadvantages and limitations of the prior art hereinabove enumerated. Another object of the present invention is to provide a novel TCP/IP protocol that does not rely on the state of a connection being maintained.

According to one embodiment of the present invention, a synchronization packet is preformed at a first device. The synchronization packet is to be transmitted to a second device through a connection over a computer network. The second device, in response to receiving the synchronization packet develops an

acknowledgment packet to be transmitted back to the first device. The second device reads selected header information from the synchronization packet and re-uses this information for the header of the acknowledgment packet. The first and second device exchange a plurality of successively transmitted packets and a plurality of acknowledgment packets. For each packet to be developed and sent, the originator of the packet similarly reads the selected header information of the immediately prior received packet and re-uses this header information for the current packet. The first device may also preform a termination packet, and the second device acknowledges the termination packet in an acknowledgment packet that re-uses header information from the termination packet.

A feature of the present invention is that by re-using header information between a received packet and the next to be sent packet, the state of a connection between a client and server need not be maintained. Another feature of the present invention is that intensive processing of the stack at each of the client and server need not be performed upon receipt and transmission of packets and acknowledgments. The protocol of the present invention is therefore highly advantageous in testing of connections and rate of forming connections in network devices.

These and other objects, advantages and features of the present invention will become readily apparent to those skilled in the art from a study of the following Description of an Exemplary Preferred Embodiment when read in conjunction with the attached Drawing and appended Claims.

Brief Description of the Drawing

Fig. 1 is a schematic diagram of a computer system useful for practicing the present invention;

Fig. 2 is a format of a packet transmitted over the network of Fig. 1;

Fig. 3 is the format of a TCP header;

Fig. 4 is the format of an IP header;

Fig. 5A-C are flowcharts useful describe a method of the present invention; and

Fig. 6 is a schematic diagram of another computer system useful to practice the present invention in a testing environment.

Description of the Exemplary Preferred Embodiments

Referring now to Fig. 1, there is shown a computer system 10. The computer system 10 includes a first device 12, a second device 14 and a computer network 16. Each of the first device 12 and the second device 14 may be any of a personal computer, a workstation, a web server or any similar type of device. The computer network 16 is, in a preferred embodiment of the present invention, a public computer network such as the Internet, however, any computer network, in which communication between the first device 12 in the second device 14 is accomplished by the exchange of successively transmitted packets, is within the scope of the present invention. One such example is a public or private, local or wide area network using Ethernet packets for communication.

The packets are transmitted bidirectionally over the network 16 through a connection established between the first device 12 and the second device 14. Typically, a program executing at the first device 12 generates the data that is

transmitted to the second device 14, and a program executing at the second device 14 utilizes such data. Conversely, the program executing at the second device 14 also generates other data that is transmitted to the first device 12, and the program executing at the first device 12 utilizes this other data. Each of the successively transmitted packets exchanged between the first device 12 and the second device 14 transmits units of the data generated by the respective programs at each of the first device 12 and the second device 14. These units of data may be measured as bytes, octets or any other convenient unit measurement of data known in the art.

With reference now to Fig. 2, there is shown an exemplary packet 18. The exemplary packet 18 includes a header 20 and a payload 22. The payload 22 contains a selected number of the units of data being transmitted from one of the first device 12 and the second device 14 to the other of the first device 12 and the second device 14. The header 20 includes a TCP header 24 and an IP header 26. As is conventionally known, the packet 18 may also include a data link or MAC layer header such as an Ethernet header 28 and trailer 30 encapsulating a datagram defined by the header 20 and payload 22.

Referring now to Fig. 3, there is shown a detail of the TCP header 24. Although the details of the TCP header 24 are well known, reference is made herein below to certain fields of the TCP header 24. Specifically, the fields in the TCP header 24 of interest herein are a source port field 32, a destination port field 34, a sequence number field 36, an acknowledgment number field 38, a checksum field 40 and a control bit field 42.

Referring also to Fig. 4, there is shown a detail of the IP header 26. Similarly, although the details of the IP header 26 are well known, reference is also made herein below to certain fields of IP header 26. Specifically, the fields in the IP

header 26 of interest herein are a length field 44, a checksum field 46, a source address field 48 and a destination address field 50.

A complete description of the TCP header 24 may be found in RFC 793, Transmission Control Protocol, DARPA Internet Program, Protocol Specification, September 1981, and a complete description of the IP header 26 may be found in RFC 791, Internet Protocol, DARPA Internet Program, Protocol Specification, September 1981. However, the present invention utilizes the above referenced fields, or generates the content for these fields, in a manner not taught, suggested or disclosed within their respective protocol specifications.

For example, the above referenced TCP/IP protocol specifications require that the TCP/IP protocol be state driven, wherein a state of the connection be maintained in each of the first device 12 and the second device 14. However, it will become apparent to those skilled in the art from the description below that the novel TCP/IP protocol of the present invention is stateless. The novel TCP/IP protocol of the present invention does not rely upon a current state of the connection being stored in memory in either the first device 12 or second device 14. Other examples will become readily apparent from the following description.

In the computer system 10, in accordance with one embodiment of the present invention, the first device 12 and the second device 14 through a connection over the computer network 16 exchange a plurality of successively transmitted packets and a plurality of acknowledgment packets, wherein each of the packets is similar to the exemplary packet 18. For each of the successively transmitted packets sent from an originating one of the first device 12 and the second device 14 and received at a receiving one of the first device 12 and the second device 14, the receiving one sends

a respective one of said acknowledgment packets to the originating one of the first device 12 and the second device 14.

In accordance with the present invention, a first one of the successively transmitted packets 18 is preformed at the first device 12. The header 20 of the first one of the packets includes predetermined information associated with establishing the connection between the first device 12 and the second device 14. The first one of the successively transmitted packets is to be transmitted to the second device 14.

In response to receiving the first one of the successively transmitted packets 18 at the second device 14, the second device 14 develops a first one of the acknowledgment packets. The header 20 of the first one of the acknowledgment packets includes complementary information associated with establishing the connection between the first device 12 and the second device 14. The first one of the acknowledgment packets is to be sent to the first device 12.

More particularly, to develop the first one of the acknowledgment packets the second device 12 reads selected portions of the header 20 of the first one of the successively transmitted packets and modifies these selected portions. The second device 14 then writes the selected portions, as modified, into the header 20 of the first one of the acknowledgment packets.

To develop the header 20 for each subsequent one of the successively transmitted packets sent by the originating one of the first device 12 and the second device 14, the originating one reads selected portions of the header 20 of a prior received one of the acknowledgment packets that has been received at the originating one in acknowledgment of an immediately prior one of the successively transmitted

packets sent by the originating one. The originating one of the first device 12 and the second device 14 then modifies the selected portions from the prior received one of the acknowledgment packets of the header 20 of this particular one of the acknowledgment packets. The originating one of the first device 12 and the second device 14 then writes the selected portions, as modified, into the header 20 of the subsequent one of the successively transmitted packets to be sent by the originating one.

To develop the header 20 for each subsequent respective one of the acknowledgment packets, the receiving one of the first device 12 and the second device 14 read selected portions of the header 20 of the current one of the successively transmitted packets. The receiving one of the first device 12 and the second device 14 then modifies the selected portions from the header 20 of the current one of the successively transmitted packets. The receiving one of the first device 12 and the second device 14 then writes the selected portions from the header 20 of the current one of the successively transmitted packets, as modified, into the header 20 of the respective one of the acknowledgment packets.

A final one of the successively transmitted packets is also preformed at the first device to be transmitted to the second device. The header 20 of the final one of the successively transmitted packets includes predetermined information associated with terminating the connection between the first device 12 and the second device 14 over the network 16.

A final one of the acknowledgment packets is developed at the second device 14 in response to receiving the final one of the successively transmitted packets. The header 20 of the final one of the acknowledgment packets includes complementary information associated with terminating this connection.

More particularly, the second device 14 reads selected portions of the header 20 of the final one of the successively transmitted packets and modifies the selected portions from the final one of the successively transmitted packets. The second device 14 then writes the selected portions from the final one of the successively transmitted packets, as modified, into the header 20 of the final one of the acknowledgment packets.

Generally, the modifications referred to above may be a swapping of the contents, or the order of such contents from the header of one packet to the header of another packet. Alternatively, such modifications may take the contents from the header of one packet and alter its contents before being written to the header of another packet. It is one general intent of the present invention that header information from a received packet be re-used to the extent possible for insertion into the header of the next to be sent packet. Specific examples of such modifications are set forth below.

The preformed first one of the successively transmitted packets may be a synchronization packet wherein the header 20 of the synchronization packet includes a predetermined sequence number in the sequence number field 36 of the TCP header 24. The setting of the SYN bit in the control bit field 42 of the TCP header 24 establishes any packet 18 as a synchronization packet. When the second device 14 receives the synchronization packet, it reads the sequence number and increments the sequence number. The second device 14 then writes the incremented sequence number to the acknowledgment number field 38 of the TCP header 24 of the header 20 of the first one of the acknowledgment packets. The setting of the ACK bit in the control bit field 42 of the TCP header 24 establishes any packet 18 as an acknowledgment packet.

The header 20 of the synchronization packet may also include a source address of the first device 12 in the source address field 48 of the IP header 26 and a destination address of the second device 14 in the destination address field 50 of the IP header 26. When the second device 14 receives the synchronization packet, it reads each of the source address and destination address from the respective source address field 48 and the destination address field 50 of the IP header 26 of the header 20 of the synchronization packet. The second device 14 then writes the source address to the destination address field 50 of the IP header 26 in the header 20 of the first one of the acknowledgment packets. Similarly, the second device 14 also writes the destination address to the source address field 48 of the IP header 26 and header 20 of the first one of the acknowledgment packets.

The header 20 of the synchronization packet may also include a source port of the program executing in the first device 12 in the source port field 32 of the TCP header 24 and a destination port of the program executing in the second device 14 in the destination port field 34 of the TCP header 24. When the second device 14 receives the synchronization packet, it reads each of the source port and destination port from the respective source port field 32 and the destination port field 34 of the TCP header 24 of the header 20 of the synchronization packet. The second device 14 then writes the source port to the destination port field 34 of the TCP header 24 in the header 20 of the first one of the acknowledgment packets. Similarly, the second device 14 also writes the destination port to the source port field 32 of the TCP header 24 in the header 20 of the first one of the acknowledgment packets.

The header 20 of the synchronization packet may also include a checksum in either of the checksum field 40 of the TCP header 24 and the checksum field 46 of the IP header 26, or in both. When the second device 14 receives the synchronization packet, it reads the checksum from the header 20 of the synchronization packet. The

second device 14 then modifies the checksum in accordance with other modifications made to the header 20 of the first one of the acknowledgment packets to obviate recalculation of the checksum. The second device 14 then writes the checksum, as modified, to the appropriate one of the checksum field 40 of the TCP header 24 and the checksum field 46 of the IP header 26 in the header 20 of the first one of the acknowledgment packets.

The preformed final one of the successively transmitted packets may be a termination packet wherein the header 20 of the termination packet includes a predetermined sequence number in the sequence number field 36 of the TCP header 24. The setting of the FIN bit in the control bit field 42 of the TCP header 24 establishes any packet 18 as a termination packet. When the second device 14 receives the termination packet, it reads the sequence number and increments the sequence number. The second device 14 then writes the incremented sequence number to the acknowledgment number field 38 of the TCP header 24 of the header 20 of the final one of the acknowledgment packets.

The header 20 of the termination packet may also include the source address of the first device 12 in the source address field 48 of the IP header 26 and the destination address of the second device 14 in the destination address field 50 of the IP header 26. When the second device 14 receives the termination packet, it reads each of the source address and destination address from the respective source address field 48 and the destination address field 50 of the IP header 26 of the header 20 of the termination packet. The second device 14 then writes the source address to the destination address field 50 of the IP header 26 in the header 20 of the final one of the acknowledgment packets. Similarly, the second device 14 also writes the destination address to the source address field 48 of the IP header 26 and header 20 of the final one of the acknowledgment packets.

The header 20 of the termination packet may also include the source port of the program executing in the first device 12 in the source port field 32 of the TCP header 24 and the destination port of the program executing in the second device 14 in the destination port field 34 of the TCP header 24. When the second device 14 receives the termination packet, it reads each of the source port and destination port from the respective source port field 32 and the destination port field 34 of the TCP header 24 of the header 20 of the termination packet. The second device 14 then writes the source port to the destination port field 34 of the TCP header 24 in the header 20 of the final one of the acknowledgment packets. Similarly, the second device 14 also writes the destination port to the source port field 32 of the TCP header 24 in the header 20 of the final one of the acknowledgment packets.

The header 20 of the termination packet may also include a checksum in either of the checksum field 40 of the TCP header 24 and the checksum field 46 of the IP header 26, or in both. When the second device 14 receives the termination packet, it reads the checksum from the header 20 of the synchronization packet. The second device 14 then modifies the checksum in accordance with other modifications made to the header 20 of the final one of the acknowledgment packets to obviate recalculation of the checksum. The second device 14 then writes the checksum, as modified, to the appropriate one of the checksum field 40 of the TCP header 24 and the checksum field 46 of the IP header 26 in the header 20 of the final one of the acknowledgment packets.

Other than the first one and the final one of the acknowledgment packets, each respective one of the acknowledgment packets is developed in response to receiving the current one of the successively transmitted packets sent by the originating one of the first device 12 and the second device 14. Similarly as described with respect to

the first one and the final one of the acknowledgment packets, each of the above fields in the header 20 of the current one of the successively transmitted packets are read, modified as the case may be, and written to the appropriate one of the header fields in the header 20 of the subsequent one of the acknowledgment packets. However, one significant difference exits with respect to developing the acknowledgment number for each respective one of the acknowledgment packets.

The header 20 of each current one of the successively transmitted packets includes the sequence number in the sequence number field of the TCP header 24 and also a length number in the length number field 44 of the IP header 26. The length number is commensurate with the number of the units of data contained within the payload 22.

In the above referenced TCP/IP protocol specifications, the length number is the total length of the datagram. Since the TCP header 24 and the IP header 26 are each of a fixed length, the length number is therefore commensurate with and determinative of the number of units of data in the payload 22.

The originating one of the first device 12 and the second device 14 reads each of the sequence number and the length number from the header 20 of the current one of the successively transmitted packets and develops the acknowledgment number for the respective one of the acknowledgment packets from the sequence number and the length number. The receiving one of the first device 12 and the second device 14 then writes this acknowledgment number to the acknowledgment number field 38 of the TCP header 24 in the header 20 of the respective one of the acknowledgment packets. When the length number refers to the number of units of data in the payload 22, the second device 14 may calculate the acknowledgment number as a sum of the sequence number and the length number.

The header 20 of the current one of the acknowledgment packets received at the originating one of the first device 12 and the second device 14 in acknowledgment of an immediately prior one of the successively transmitted packets includes an acknowledgment number in the acknowledgment number field 38 of the TCP header 24. The originating one of the first device 12 and a second device 14 developing each subsequent one of the successively transmitted packets reads the acknowledgment number from the header 20 of the current one of the acknowledgment packets and develops a sequence number from the acknowledgment number. The originating one then writes the sequence number into the sequence number field 36 of the TCP header 24 for the subsequent one of the successively transmitted packets being developed. To develop the sequence number, the acknowledgment number may, in one embodiment of the present invention, be incremented.

The header 20 of the current one of the acknowledgment packets received in acknowledgment of the immediately prior one of the successively transmitted packets may also include the source address of the originating one of the first device 12 and the second device 14 in the destination address field 50 of the IP header 26 and the destination address of the receiving one of the first device 12 and the second device 14 in the source address field 48 of the IP header 26. The originating one of the first device 12 and the second device 14 developing the subsequent one of the successively transmitted packets reads the source address and the destination address and writes the source address to the source address field 48 of the IP header 26 of the subsequent one of the successively transmitted packets and writes the destination address to the destination address field 50 of the IP header 26 of the subsequent one of the successively transmitted packets.

The header 20 of the current one of the acknowledgment packets received in acknowledgment of the immediately prior one of the successively transmitted packets

may also include the source port of the program executing at the originating one of the first device 12 and the second device 14 in the destination port field 34 of the TCP header 24 and the destination port of the program running at the receiving one of the first device 12 and the second device 14 in the source port field 32 of the TCP header 24. The originating one of the first device 12 and the second device 14 developing each subsequent one of the successively transmitted packets reads the source port and the destination port and writes the source port to the source port field 32 of the TCP header 24 of the subsequent one of the successively transmitted packets and writes the destination port to the destination port field 34 of the TCP header 24 of the subsequent one of the successively transmitted packets.

The header 20 of the current one of the acknowledgment packets received in acknowledgment of the immediately prior one of the successively transmitted packets may also include a checksum in either of the checksum field 40 of the TCP header 24 and the checksum field 46 of the IP header 26, or in both. The originating one of the first device 12 and the second device 14 reads the checksum from the header 20 of the current one of the acknowledgment packets. The originating one then modifies the checksum in accordance with other modifications made to the header 20 of the subsequent one of the successively transmitted packets to obviate recalculation of the checksum. The originating one then writes the checksum, as modified, to the appropriate one of the checksum field 40 of the TCP header 24 and the checksum field 46 of the IP header 26 in the header 20 of the subsequent one of the successively transmitted packets.

In the computer system 10, in another embodiment of the present invention, wherein a program executing at the first device 12 and a program executing at the second device 14 exchange data in a plurality of packets 18 transmitted bidirectionally through a connection over the computer network 16 and further wherein each of the

packets 18 has the header 20 and the payload 22, the first device 12 may predetermine a total first number of units of data in the payload 22 of all the packets 18 to be transmitted from the first device 12 to the second device 14. Similarly, the second device 14 may predetermine a total second number of units of data in the payload 22 of all the packets 18 to be transmitted from the second device 14 to the first device 12.

The first device 12 preforms a SYN packet to be transmitted to the second device 14 as a first one of the packets 18. The header 20 of the SYN packet includes an initial sequence number associated with the first device 12 in the sequence number field 36 of the TCP header 24.

The first device 12 also preforms a FIN packet as a final one of the packets 18 to be transmitted to the second device 14. The header 20 of the FIN packet includes a final sequence number associated with the first device 12 in the sequence number field 36 of the TCP header 24. The final sequence number is determinable from the first number of units of data.

The second device 14 develops a SYN/ACK packet as a further one of the bidirectionally transmitted packets 18 in response to receipt of the SYN packet. The header 20 of the SYN/ACK packet includes an acknowledgment number in the acknowledgment number field 38 of the TCP header 24. The acknowledgment number is determinable from the initial sequence number associated with the first device 12. The header 20 of the SYN/ACK packet also includes an initial sequence number associated with the second device 14 in the sequence number field 36 of the TCP header 24.

The second device 14 also develops a FIN/ACK packet as a final one of the bidirectionally transmitted packets 18 in response to receipt of the FIN packet. The header 20 of the FIN/ACK packet includes an acknowledgment number in the acknowledgment number field 38 of the TCP header 24 that is determinable from the final sequence number in the header 20 of the FIN packet. The header 20 of the FIN/ACK packet also includes a final sequence number associated with the second device 14 in the sequence number field 36 of the TCP header 24. The final sequence number is determinable from the second number of units of data.

For a SYN packet, the SYN bit is set within the control bit field 42 of the TCP header 24. For a SYN/ACK packet, each of the SYN and ACK bits are set within the control bit field 42 of the TCP header 24. For a FIN packet, the FIN bit is set within the control bit field 42 of the TCP header 24. For a FIN/ACK packet, each of the FIN and ACK bits are set within the control bit field 42 of the TCP header 24.

To develop the SYN/ACK packet the second device 14 reads the initial sequence number associated with the first device 12 upon receipt of the SYN packet. The second device then increments the initial sequence number associated with the first device 12 to develop the acknowledgment number for the SYN/ACK packet and writes this acknowledgment number to the acknowledgment number field 38 of the TCP header 24 of the SYN/ACK packet.

To develop the FIN/ACK packet the second device 14 reads the final sequence number associated with the first device 12 upon receipt of the FIN packet. The second device then increments the final sequence number associated with the first device 12 to develop the acknowledgment number for the FIN/ACK packet and writes

this acknowledgment number to the acknowledgment number field 38 of the TCP header 24 of the FIN/ACK packet.

The first device 12 may determine the final sequence number of the FIN packet as a sum of the initial sequence number associated with the first device 12 and the first number of units of data. Similarly, the second device 14 may determine the final sequence number of the FIN/ACK packet and sum of the final sequence number associated with the second device 14 and the second number of units of data.

For each of the SYN packet, the SYN/ACK packet, the FIN packet and the FIN/ACK packet, the source port field 32, the destination port field 34, the sequence number field 36, the acknowledgment number field 38 and the checksum field 40 of the TCP header 24 have their contents read, modified and written as described hereinabove with respect to the first one of the successively transmitted packets, the first one of the acknowledgment packets, the final one of the successively transmitted packets and the final one of the acknowledgment packets, respectively.

Similarly, for each of the SYN packet, the SYN/ACK packet, the FIN packet and the FIN/ACK packet, the length field 44, the checksum field 46, the source address field 48 and the destination address field 50 in the IP header 26 have their contents read, modified and written as described hereinabove with respect to the first one of the successively transmitted packets, the first one of the acknowledgment packets, the final one of the successively transmitted packets and final one of the acknowledgment packets, respectively.

In the one aspect of the present embodiment of the present invention, the header for each one of the packets 18 sent from an originating one of the first device 12 and the second device 14 may be developed as hereinabove described from the

header of the current one of the acknowledgment packets sent from the receiving one of the first device 12 and the second device 14 knowledge in receipt of an immediately prior one of the packets 18 sent from the originating one. Similarly, the header for each one of the acknowledgment packets may also be developed as hereinabove described from the header of the respective one of the packets 18 received at the receiving one of the first device 12 and second device 14.

In another aspect of the present embodiment of the present invention, each of packets 18 sent from one of the first device 12 and the second device 14 may contain data in the payload 22 and also acknowledge another one of the packets 18 received from the other one of the first device 12 and the second device 14. Each of the packets 18 containing data in the payload 22 and also acknowledging another one of the packets 18 received has the ACK bit set within the control bit field 42 of the TCP header 24.

For example, the header 20 for a current one of the packets 18 to be sent by one of the first device 12 and the second device 14 may include an acknowledgment number in the acknowledgment number field 38 of the TCP header 24, a sequence number in the sequence number field 36 of the TCP header 24 and a length number in the length number field 44 of the IP header 26. When the current one of the packets 18 is received at the other one of the first device 12 and the second device 14, a next one of the packets 18 is then developed at the other, or receiving, one of the first device 12 and a second device 14.

The receiving one of the first device 12 and the second device 14 reads each of the acknowledgment number, the sequence number and the length number from the header 20 of the received current one of the packets 18. The receiving one of the first device 12 and the second device 14 develops the sequence number for the next

one of the packets 18 from the acknowledgment number read from the current one of the packets 18. Similarly, the receiving one of the first device 12 and the second device 14 develops the acknowledgment number for the next one of the packets 18 from the sequence number in the length number from the current one of the packets 18.

The receiving one of the first device 12 and the second device 14 then writes the developed sequence number and the developed acknowledgment number for the next one of the packets 18 into the sequence number field 36 and acknowledgment number field 38, respectively, of the TCP header 24 of the next one of the packets 18. The receiving one of the first device 12 and the second device 14 also develops the length number for the length number field 44 in the IP header 26 for the next one of the packets 18 commensurately with the number of units of data in the payload 22 in the next one of the packets 18.

In the computer system 10, and another embodiment of the present invention, first successive ones of the packets 18 are sent from the first device 12 to the second device 14 and second successive ones of the packets 18 are sent from the second device 14 to the first device 12. Each of the first successive ones and second successive ones of the packets 18 include the header 20 and the payload 22 so that data is exchanged between the first device 12 and second device 14 between programs executing at each of the first device 12 and the second device 14. The header 20 of each of the first successive ones and second successive ones of the packets 18 includes the sequence number in the sequence number field 36 of the TCP header 24, and acknowledgment number in the acknowledgment number field 38 of the TCP header 24, and the length number in the length number field 44 of the IP header 26.

At the first device 12, a SYN packet is preformed to be transmitted to the second device 14. The SYN packet has an initial sequence number associated with the first device 12 in its sequence number field 36. The second device 14 in response to receipt of the SYN packet develops a SYN/ACK packet to be transmitted to the first device 12. The header 20 of the SYN/ACK packet includes an acknowledgment number in the acknowledgment number field 38 and initial sequence number associated with the second device 14 in the sequence number field 36. The acknowledgment number is determinable from the initial sequence number associated with the first device 12.

At the first device 12, a FIN packet is also preformed to be transmitted to the second device 14. The FIN packet has a final sequence number associated with the first device 12 in its sequence number field 36. The final sequence number is determinable from said first number of units of data. The second device 14 in response to receipt of the FIN packet develops a FIN/ACK packet to be transmitted to the first device 12. The header 20 of the SYN/ACK packet includes an acknowledgment number in the acknowledgment number field 38 and a final sequence number associated with the second device 14 in the sequence number field 36. The acknowledgment number is determinable from the initial sequence number associated with the first device 12. The final sequence number is determinable from said number of said second units of data.

For each current one of the first packets to be sent from the first device 12, the first device 12 reads the sequence number, the length number and the acknowledgment number from the header 20 of a respective one of the second packets received at the first device 12 subsequently to an immediately prior one of the first packets being sent by the first device 12. The first device 12 develops for the current one of the first packets each of a current acknowledgment number from the sequence

number and the length number from the header 20 of the respective one of the second packets, and a current sequence number from the acknowledgment number of the respective one of the second packets. The first device 12 then writes the current acknowledgment number and the current sequence number to the acknowledgment number field 38 and the sequence number field 36, respectively, of the header 20 of the current one of the first packets.

Similarly, for each current one of the second packets to be sent from the second device 14, the second device 14 reads the sequence number, the length number and the acknowledgment number from the header 20 of a respective one of the first packets received at the second to device 14 subsequently to an immediately prior one of the second packets being sent by the second device 14. The second device 14 develops for the current one of the second packets each of a current acknowledgment number from the sequence number and the length number from the header 20 of the respective one of the first packets, and a current sequence number from the acknowledgment number of the respective one of the first packets. The second device 14 then writes the current acknowledgment number and the current sequence number to the acknowledgment number field 38 and the sequence number field 36, respectively, of the header 20 of the current one of the second packets.

The description of each of the header fields in the header 20 of the first and second packets, the SYN packets, the SYN/ACK packet, the FIN packet and the FIN/ACK packet, as well as the contents of these header fields, are read, modified and written as described above. Such description is accordingly incorporated into description of the presently described embodiment of the invention. Additionally, the setting of an ACK bit in the control field 42 may occur contemporaneously with the writing of the acknowledgment number in the acknowledgment number field 38.

Referring now to Fig. 5A, there is shown a flowchart useful to describe a method of the present invention. In one embodiment of the present invention, the method may be practice in conjunction with a testing environment in which the first device 12 and the second device 14 communicate with each other, similarly as described above, however, with a device under test (DUT) 58 being interposed a first device 12 and the second device 14 within the network 16, as best seen in Fig. 6. The DUT 58 may be any conventional network device, such as a server, router or the like.

For example, the program executing at the first device 12 and the program executing at the second device 14 may be testing routines to determine a maximum rate of establishing connections at or through the device under test 58, or the maximum number of connections possible to be maintained at the device under test 58. The data in the payload 22 of each one of the packets 18 may then be data generated by these testing routines to be transmitted between the first device 12 and the second device 14. More specifically, the data generating at the originating one of the first device 12 in the second device 14 and placed into the payload 22 of the packets 18 originating therefrom may be analyzed at the receiving one of the first device 12 in the second device 14 for to determine the empirical results of the effect of establishing and maintaining connections at the DUT 58.

In accordance with the principles of one embodiment of the present invention, a stateless TCP/IP method includes predetermining, at step 60, a total number of packets or units as data to be sent between the first device 12 and a second device 14. In one particular embodiment of the present invention, a total first number of units of data in the payload 22 of all packets to be transmitted from the first device 12 to the second device 14 and a total of the second member units of data in the payload 22 of all the packets 18 to be transmitted from the second device 14 to the first device

12. The total number of first and second packets is determined for each connection established between the first device 12 and the second device 14. In the testing environment, these connections are also through the device under test 58.

At step 62, the method further includes preforming a SYN packet at the first device 12. The SYN packet is to be transmitted to the second device 14 as a first one of the packets. The header 20 of the SYN packet includes and initial sequence number associated with the first device 12 in the sequence number field 36 of the TCP header 24. The initial sequence member may be randomly generated or may further be preselected.

At step 64, the method further includes preforming a FIN packet at the first device 12 as a final one of the packets 18 to be transmitted to the second device 18. The FIN packet is to be transmitted to the second device 14 as a final one of the packets. The header 20 of the FIN packet includes a final sequence number associated with the first device in the sequence number field 36 of the TCP header 24. The final sequence number is determinable from the first number of units of data and the initial sequence number associated with the first device 12.

In response to receipt of the SYN packet at the second device 14, the method includes, at step 66, developing a SYN/ACK packet at the second device 12. The header 20 of the SYN/ACK packet includes an acknowledgment number in the acknowledgment number field 38 of the TCP header 24. The header 20 of the SYN/ACK packet further includes an initial sequence number associated with the second device 14 in the sequence number field 36 of the TCP header 24. The acknowledgment number in the SYN/ACK packet is determinable from the initial sequence number associated with the first device 12. The initial sequence number

associated with the second device 14 may also be randomly generated or predetermined.

At step 68, is indicated that the first ones of the packets 18 are transmitted from the first device 12 to the second device 14 and the second ones of the packets 18 are transmitted from the second device 14 to the first device 12. Step 68 is described in greater detail hereinbelow with reference to Fig.'s 5B and 5C.

At step 70, in response to receipt of the FIN packet at the second device 14, the method includes developing a FIN/ACK packet at the second device 14 as a final one of the packets. The header 20 of the FIN/ACK packet includes an acknowledgment number determinable from the final sequence number associated with the first device 12 in the acknowledgment number field 38 of the TCP header 24 and a final sequence number associated with the second device 14 in the sequence number field 36 of the TCP header 24. The final sequence number associated with the second device is determinable from the second number of units of data and the initial sequence number associated with the second device 14.

In conjunction with preforming the test at the DUT 58, the above described method of Fig. 5A would be repeatedly executed at one of the first device 12 and the second device 14 to establish as many connections possible at the DUT 58.

In one embodiment of the present invention, the above described method made that predetermined the total number of units of data or the total number of packets to be transmitted between the first device 12 in the second device 14. Accordingly, the final sequence number associated with the first device 12 and the second device 14 would need to be determined from their respective initial sequence number and the number of units of data actually transmitted from each device.

The header 20 of each of the packets 18 have the sequence number field 36 containing a sequence number of a current one of the packets, a length number field 44 containing a length number of the units of data contained in the payload 22 for such packet, and an acknowledgment number field 38 containing an acknowledgment number.

With reference to Fig. 5B, with respect to a current one of the first packets to be sent, the method may further include reading the sequence number, the length number and the acknowledgment number from the header 20 of a respective one of the second packets received at the first device 12 subsequently to an immediately prior one of the first packets being sent, as indicated at step 72. The method of may further include, at step 74, developing for the current one of the first packets, each of a current acknowledgment number from the sequence number and the length number from the header of the respective one of the second packets, and a current sequence number from the acknowledgment number of the respective one of the second packets. The method may then include, at step 76, writing the current acknowledgment number and the current sequence number to the acknowledgment number field 38 and the sequence number field 36, respectively, of the header 20 of the current one of the first packets.

With reference to Fig. 5C, with respect to a current one of the second packets to be sent, the method may further include reading the sequence number, the length number and the acknowledgment number from the header 20 of a respective one of the first packets received at the second device 14 subsequently to an immediately prior one of the second packets being sent, as indicated at step 78. The method of may further include, at step 80, developing for the current one of the second packets, each of a current acknowledgment number from the sequence number and the length number from the header of the respective one of the first packets, and a current

sequence number from the acknowledgment number of the respective one of the first packets. The method may then include, at step 82, writing the current acknowledgment number and the current sequence number to the acknowledgment number field 38 and the sequence number field 36, respectively, of the header 20 of the current one of the second packets.

In the method of the present invention, it is apparent that the header 20 for each of the current first or second packets to be developed uses, to the extent possible, as much information is possible from the immediately prior received one of the packets at either the first device 12 of the second device 14. This statement also holds true for the SYN/ACK packet and the FIN/ACK packet.

For example, in addition to the elements of the header 20 described above, any of the above described packets may have the source address of the one of the first device 12 and the second device 14 developing such packet in the source address field 48 and a destination address of the other one of the first device 12 in the second device 14 in the destination address field 50. Upon this packet being received, the receiving device reads each of the source address and the destination address from the received packet and writes this information, respectively, to the destination address field 50 and the source address field 48 of the new packet to be developed. This statement also holds true for the contents of the source port field 32 and destination port field 34.

Regarding the contents of the checksum field 40 in the TCP header 24 and also the checksum field 46 of the IP header 26, the checksum field is read at the receiving device. However, instead of recalculating the checksum using known checksum algorithms, the checksum may be artificially modified, based upon the new values in the header. For example, it is known that incrementing of a sequence

number to develop the new acknowledgment number would have a corresponding effect at the checksum. For example, for each binary 1 added to a sequence number to form an acknowledgment number, a binary one may be subtracted from the read checksum to develop the new checksum. In such way, recalculation of the checksum has been obviated.

There is described hereinabove a novel TCP/IP method and apparatus. Those skilled in the art may not make numerous uses of, and departures from, the above described preferred embodiments without departing from the inventive concepts disclosed herein. Accordingly, the present invention is to be defined solely by the scope of the appended Claims.